UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/677,744	10/03/2003	Michael C. Dapp	T3497-9109US01	4899

181 7590 03/05/2008
MILES & STOCKBRIDGE PC
1751 PINNACLE DRIVE
SUITE 500
MCLEAN, VA 22102-3833

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2192	
NOTIFICATION DATE	**DELIVERY MODE**
03/05/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

ipdocketing@milesstockbridge.com
sstiles@milesstockbridge.com

Office Action Summary	**Application No.** 10/677,744	**Applicant(s)** DAPP ET AL.	
	Examiner Michael J. Yigdall	**Art Unit** 2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *10 December 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *26-34* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *26-34* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date *10/24/07*.

4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1. This Office action is responsive to Applicant's submission filed on December 10, 2007. Claims 26-34 are pending.

Response to Amendment

2. The objection to the specification has been withdrawn in view of Applicant's amendment.

3. The objection to claims 28-30, 32 and 34 has been withdrawn in view of Applicant's amendment.

Response to Arguments

4. Applicant's arguments have been considered but are moot in view of the new ground(s) of rejection, as set forth below. Applicant's amendment necessitated the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 26-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 7,188,168 to Liao (art of record, "Liao") in view of U.S. Patent No. 6,141,792 to Acker et al.

(now made of record, “Acker”) and in view of U.S. Pub. No. 2004/0073870 to Fuh et al. (art of record, “Fuh”).

With respect to claim 26 (currently amended), Liao teaches a parser accelerator apparatus having an adaptive learning capability (see, for example, column 5, lines 4-15, which shows a programmable hardware parser apparatus), the apparatus comprising:

a finite state machine configured to parse a document (see, for example, FIG. 3 and column 13, lines 62-66, which shows a finite state machine for parsing an input stream);

a memory configured to store at least one state table (see, for example, column 13, lines 52-55, which shows memory for storing a data structure that represents a state table);

a token buffer configured to store at least one token received from the finite state machine (see, for example, column 14, lines 19-22, which shows that the finite state machine recognizes tokens, and column 15, lines 35-38, which shows a buffer for storing such tokens);

a parser accelerator compiler configured to compile a grammar specification and to generate a state transition specification (see, for example, compiler 102 in FIG. 1 and column 8, lines 14-22, which shows a parser compiler for compiling a GPCL program and generating a state transition specification, and column 7, lines 16-38, which shows that the GPCL program comprises a grammar specification).

Liao does not expressly disclose the parser accelerator compiler comprising a plurality of object classes including at least a main class and an input manager class having a production rule override section for re-specifying production rules.

However, in an analogous art, Acker teaches a compiler (see, for example, FIG. 3) that comprises a plurality of object classes (see, for example, column 5, lines 40-44). The object classes include at least a main class in the form of an “emitter class” (see, for example, column 13, lines 2-6) and input manager classes in the form of “entry classes” (see, for example, column 7, lines 15-23). Acker further teaches re-specifying production rules with user-defined subclasses that override the production rules of the entry classes, even though the actual entry classes are not open to the programmer (see, for example, column 6, lines 6-16).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the parser accelerator compiler comprises a plurality of object classes including at least a main class and an input manager class having a production rule override section for re-specifying production rules, as Acker suggests, so as re-specify the production rules without changing the actual production rules. One of ordinary skill in the art could have implemented the compiler of Liao as a plurality of object classes with predictable results.

Liao does not expressly disclose that the state transition specification is generated in a self describing format.

However, in an analogous art, Fuh teaches an apparatus comprising a compiler, a parser and a loader (see, for example, FIGS. 2 and 10, and paragraphs [0031] and [0050]). Fuh teaches that the parser employs a state transition table (see, for example, paragraph [0039], lines 16-18) that corresponds to a state transition specification in the form of an XML schema definition (see, for example, paragraph [0034]). Thus, the state transition specification is written in a self-describing format (see, for example, FIG. 4). Such a format is well known in the art and facilitates interoperability (see, for example, paragraph [0003]).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the state transition specification is generated in a self describing format, as Fuh suggests, so as to facilitate interoperability.

Liao in view of Acker and Fuh further teaches or suggests:

a parser accelerator loader configured to load the memory with a state table corresponding to the state transition specification (see, for example, Liao, column 13, lines 55-56, which shows a loader for loading the memory with the data structure that represents a state table corresponding to the state transition specification),

wherein the parser accelerator compiler and the parser accelerator loader are configured to reconfigure the parser accelerator in response to detected data patterns in an electronic document in substantially real time, thereby adapting over time to provide a learning capability (see, for example, Liao, column 19, lines 40-47, which shows that the apparatus is reconfigured in substantially real time, and column 8, lines 23-28, which shows that the parser compiler detects data patterns in an electronic document to reconfigure the apparatus).

With respect to claim 27 (currently amended), Liao teaches a method for dynamically reconfiguring a parser accelerator (see, for example, column 5, lines 4-15, which shows a programmable hardware parser apparatus, and column 19, lines 40-47, which shows dynamically reconfiguring the apparatus) comprising:

providing a grammar specification (see, for example, column 6, lines 1-12, which shows providing a grammar specification in the form of GPCL);

providing a parser accelerator having a finite state machine and a state table memory (see, for example, FIG. 3 and column 13, lines 52-55 and 62-66, which shows a parser comprising a finite state machine and state table memory);

compiling the grammar to generate finite automata (see, for example, FIG. 1 and column 8, lines 14-22, which shows compiling a GPCL program to generate finite automata).

Liao does not expressly disclose the compiling further comprising re-specifying production rules in accordance with a production rules override section of an object class.

However, in an analogous art, Acker teaches a compiler (see, for example, FIG. 3) that comprises a plurality of object classes (see, for example, column 5, lines 40-44). The object classes include at least a main class in the form of an “emitter class” (see, for example, column 13, lines 2-6) and input manager classes in the form of “entry classes” (see, for example, column 7, lines 15-23). Acker further teaches re-specifying production rules with user-defined subclasses that override the production rules of the entry classes, even though the actual entry classes are not open to the programmer (see, for example, column 6, lines 6-16).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the compiling further comprises re-specifying production rules in accordance with a production rules override section of an object class, as Acker suggests, so as re-specify the production rules without changing the actual production rules. One of ordinary skill in the art could have implemented the compiler of Liao as a plurality of object classes with predictable results.

Liao in view of Acker further teaches or suggests:

creating a finite state machine state transition specification from the finite automata (see, for example, column 9, lines 26-34, which shows that a state transition specification is created from the finite automata).

Liao does not expressly disclose that the finite state machine state transition specification is created in a self describing format.

However, in an analogous art, Fuh teaches an apparatus comprising a compiler, a parser and a loader (see, for example, FIGS. 2 and 10, and paragraphs [0031] and [0050]). Fuh teaches that the parser employs a state transition table (see, for example, paragraph [0039], lines 16-18) that corresponds to a state transition specification in the form of an XML schema definition (see, for example, paragraph [0034]). Thus, the state transition specification is written in a self-describing format (see, for example, FIG. 4). Such a format is well known in the art and facilitates interoperability (see, for example, paragraph [0003]).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the finite state machine state transition specification is created in a self describing format, as Fuh suggests, so as to facilitate interoperability.

Liao in view of Acker and Fuh further teaches or suggests:

loading the finite state machine state transition specification into the state table memory (see, for example, Liao, column 13, lines 55-56, which shows loading a data structure that represents the state transition specification into the state table memory).

With respect to claim 28 (currently amended), the rejection of claim 27 is incorporated, and Liao in view of Acker and Fuh further teaches or suggests that the self-describing format is a mark-up language (see, for example, Fuh, paragraph [0003], which shows extensible mark-up language).

With respect to claim 29 (currently amended), the rejection of claim 28 is incorporated, and Liao in view of Acker and Fuh further teaches or suggests that the mark-up language is extensible mark-up language (XML) (see, for example, Fuh, paragraph [0003], which shows extensible mark-up language or XML).

With respect to claim 30 (currently amended), the rejection of claim 27 is incorporated, and Liao in view of Acker and Fuh further teaches or suggests that the grammar specification includes a specification of desirable performance characteristics of the parser accelerator (see, for example, Liao, column 6, lines 1-12, which shows that the grammar specification includes performance rules and policies for the apparatus).

With respect to claim 31 (currently amended), Liao teaches a parser accelerator apparatus having an adaptive learning capability (see, for example, column 5, lines 4-15, which shows a programmable hardware parser apparatus), the apparatus comprising:

a finite state machine configured to parse a document (see, for example, FIG. 3 and column 13, lines 62-66, which shows a finite state machine for parsing an input stream);

a memory configured to store at least one state table (see, for example, column 13, lines 52-55, which shows memory for storing a data structure that represents a state table);

a parser accelerator compiler configured to compile a grammar specification and to generate a state transition specification (see, for example, compiler 102 in FIG. 1 and column 8, lines 14-22, which shows a parser compiler for compiling a GPCL program and generating a state transition specification, and column 7, lines 16-38, which shows that the GPCL program comprises a grammar specification).

Liao does not expressly disclose the parser accelerator compiler comprising a plurality of object classes including at least a main class and an input manager class having a production rule override section for re-specifying production rules.

However, in an analogous art, Acker teaches a compiler (see, for example, FIG. 3) that comprises a plurality of object classes (see, for example, column 5, lines 40-44). The object classes include at least a main class in the form of an “emitter class” (see, for example, column 13, lines 2-6) and input manager classes in the form of “entry classes” (see, for example, column 7, lines 15-23). Acker further teaches re-specifying production rules with user-defined subclasses that override the production rules of the entry classes, even though the actual entry classes are not open to the programmer (see, for example, column 6, lines 6-16).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the parser accelerator compiler comprises a plurality of object classes including at least a main class and an input manager class having a production rule override section for re-specifying production rules, as Acker suggests, so as re-specify the production rules without changing the actual production rules. One of ordinary skill in the art could have implemented the compiler of Liao as a plurality of object classes with predictable results.

Liao does not expressly disclose that the state transition specification is generated in a self describing format comprising a mark-up language.

However, in an analogous art, Fuh teaches an apparatus comprising a compiler, a parser and a loader (see, for example, FIGS. 2 and 10, and paragraphs [0031] and [0050]). Fuh teaches that the parser employs a state transition table (see, for example, paragraph [0039], lines 16-18) that corresponds to a state transition specification in the form of an XML schema definition (see, for example, paragraph [0034]). Thus, the state transition specification is written in a self-describing format comprising a mark-up language (see, for example, FIG. 4). Such a format is well known in the art and facilitates interoperability (see, for example, paragraph [0003]).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the state transition specification is generated in a self describing format comprising a mark-up language, as Fuh suggests, so as to facilitate interoperability.

Liao in view of Acker and Fuh further teaches or suggests:

a parser accelerator loader configured to load the memory with a state table corresponding to the state transition specification (see, for example, Liao, column 13, lines 55-56, which shows a loader for loading the memory with the data structure that represents a state table corresponding to the state transition specification),

wherein the parser accelerator compiler and the parser accelerator loader are configured to reconfigure the parser accelerator in response to a changed condition (see, for example, Liao, column 19, lines 40-47, which shows reconfiguring the apparatus, and column 8, lines 23-28, which shows that the apparatus is reconfigured in response to a changed condition such as a pattern in a document).

With respect to claim 32 (currently amended), the rejection of claim 31 is incorporated, and Liao in view of Acker and Fuh further teaches or suggests that the changed condition comprises a pattern in a document (see, for example, Liao, column 8, lines 23-28, which shows that the apparatus is reconfigured in response to a pattern in a document).

7. Claim 33 is rejected under 35 U.S.C. 103(a) as being unpatentable over Liao in view of Acker.

With respect to claim 33 (currently amended), Liao teaches a method for dynamically reconfiguring a parser accelerator (see, for example, column 5, lines 4-15, which shows a programmable hardware parser apparatus, and column 19, lines 40-47, which shows dynamically reconfiguring the apparatus) comprising:

electronically providing a grammar specification including a specification of desirable performance characteristics of the parser accelerator (see, for example, column 6, lines 1-12, which shows providing a grammar specification in the form of GPCL that specifies performance rules and policies for the apparatus);

providing a parser accelerator having a finite state machine and a state table memory (see, for example, FIG. 3 and column 13, lines 52-55 and 62-66, which shows a parser comprising a finite state machine and state table memory);

compiling the grammar to generate finite automata (see, for example, FIG. 1 and column 8, lines 14-22, which shows compiling a GPCL program to generate finite automata).

Liao does not expressly disclose the compiling further comprising re-specifying production rules in accordance with a production rules override section of an object class.

However, in an analogous art, Acker teaches a compiler (see, for example, FIG. 3) that comprises a plurality of object classes (see, for example, column 5, lines 40-44). The object classes include at least a main class in the form of an "emitter class" (see, for example, column 13, lines 2-6) and input manager classes in the form of "entry classes" (see, for example, column 7, lines 15-23). Acker further teaches re-specifying production rules with user-defined subclasses that override the production rules of the entry classes, even though the actual entry classes are not open to the programmer (see, for example, column 6, lines 6-16).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that the compiling further comprises re-specifying production rules in accordance with a production rules override section of an object class, as Acker suggests, so as re-specify the production rules without changing the actual production rules. One of ordinary skill in the art could have implemented the compiler of Liao as a plurality of object classes with predictable results.

Liao in view of Acker further teaches or suggests:

creating a finite state machine state transition specification from the finite automata (see, for example, column 9, lines 26-34, which shows that a state transition specification is created from the finite automata); and

loading the finite state machine state transition specification into the state table memory (see, for example, column 13, lines 55-56, which shows loading a data structure that represents the state transition specification into the state table memory).

8. Claim 34 is rejected under 35 U.S.C. 103(a) as being unpatentable over Liao in view of Acker, as applied to claim 33 above, and further in view of Fuh.

With respect to claim 34 (currently amended), the rejection of claim 33 is incorporated. Liao does not expressly disclose that creating a finite state machine state transition specification from the finite automata comprises creating a finite state machine state transition specification in extensible mark-up language (XML).

However, in an analogous art, Fuh teaches an apparatus comprising a compiler, a parser and a loader (see, for example, FIGS. 2 and 10, and paragraphs [0031] and [0050]). Fuh teaches that the parser employs a state transition table (see, for example, paragraph [0039], lines 16-18) that corresponds to a state transition specification in the form of an XML schema definition (see, for example, paragraph [0034]). Thus, the state transition specification is written in extensible mark-up language or XML (see, for example, FIG. 4). Such a language is well known in the art and facilitates interoperability (see, for example, paragraph [0003]).

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to implement Liao such that creating a finite state machine state transition specification from the finite automata comprises creating a finite state machine state transition specification in extensible mark-up language (XML), as Fuh suggests, so as to facilitate interoperability.

Conclusion

9. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

10. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is 571-272-3707. The examiner can normally be reached on Monday to Friday from 8:00 AM to 4:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Michael J. Yigdall
Examiner
Art Unit 2192

/MY/

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192